計算機アーキテクチャ　80－9
（１９９０．１．２５）

意味ネットワークマシンＩＸＭにおける並列連想記憶

国分明男　樋口哲也　古谷立美
電子技術総合研究所

並列連想記憶のアーキテクチャ、意味ネットワークマシンＩＸＭの構成、連想メモリ上の並列演算性能について述べる。並列連想記憶は、連想メモリ上のＳＩＭＤ的並列処理の有効性の限界を解決するために提案した、連想メモリを備えるＰＥの多重構成である。

意味ネットワークマシンＩＸＭ２は、並列連想記憶構成の実現例であり、トランスピュータと４Ｋ語（１語４０ビット）の連想メモリを備える連想プロセッサ６４台と、トランスピュータと２Ｋ語の連想メモリを備え、ブロードキャスト機能を持つネットワークプロセッサ９台から構成される。連想メモリの総容量は、２５６Ｋ語（連想処理用）＋１８Ｋ語（マーカールーティング用）まで実装可能である。ＩＸＭ２の結合形態は、ローカルな完全結合をベースとした階層的接続である。この接続方式は、完全結合による通信オーバヘッドの軽減と大規模化の容易さを狙いとしている。

連想メモリ上の演算性能では、各語中の二つのデータを加算する場合に、トランスピュータによる順次加算の演算時間が語数に比例するのに対して、連想メモリによる並列加算では演算時間が一定であることを示す。１Ｋ語の各語中の８ビットデータ同士の加算では、トランスピュータによる順次加算時間が５５３０μｓであるのに対して、連想メモリによる並列加算時間は４６μｓであるという実測結果である。

Parallel Associative Memory System of Semantic Network Machine IXM

Akio Kokubu, Tetsuya Higuchi and Tatsumi Furuya
Electrotechnical Laboratory
1-1-4 Umezono, Tsukuba, Ibaraki 305, Japan

Architecture of a parallel associative memory system and configuration of the semantic network machine IXM2 are described. Performance of parallel addition with associative memory is described.

The parallel associative memory system is a multiple configuration of PEs with associative memories to support partition of a sequential problem into many sub-problems and to process them with associative memories in parallel. The semantic network machine IXM2 is an example of such parallel associative memory systems. It consists of 64 transputers with 4K words (40 bits per word) for associative processing and 9 transputers with 2K words for marker routing. The connection topology of IXM2 is a hierarchically organized local complete connection.

It is shown that bit-serial algorithms for associative memories take very short execution times than sequential algorithms for conventional computers. For 8-bit addition of 1K words, the associative memory takes only 46μs and the transputer 5530μs with IXM2.

1．はじめに

近年、半導体ＬＳＩ技術の進歩によって、大容量の連想メモリが作られるようになってきており 、今後の進展も期待できる状況にある［1-3］。

連想メモリを用いる連想処理は、連想メモリの語数によらず処理時間が一定であるというＳＩＭＤ的並列処理が可能であるため、１９６０年代には連想メモリ上の多くの論理演算アルゴリズムが研究開発された［4］。連想メモリ上のＳＩＭＤ処理は、論理演算だけでなく、リスト処理、コンピュータグラフィックス、画像認識、データベース等の多くの応用においても有用性が期待されている［5-6］。特に、各データ要素の大きさは小さくそれに対する操作は単純であるが、並列処理の程度が非常に大きいことを要求される超並列処理の分野では、大容量連想メモリを用いるアプローチは魅力的であると考えられる。

連想メモリによる処理には、問題点もある。意味ネットワークにおけるマーカー伝搬や双方向グラフにおける接続性検査などの処理に対しては、連想メモリによるＳＩＭＤ処理が必ずしも有効でなく順次処理にならざるを得ない部分が存在し、それが連想メモリによる処理のボトルネックになる。そのようなボトルネックを解決するために、意味ネットワークマシンＩＸＭ［7-9］において、並列連想記憶構成の採用を考えてきた。並列連想記憶構成では、意味ネットワークにおけるマーカー伝搬に対して等価ノード概念を導入することにより、多重マッチを複数の連想プロセッサユニットに分割割り付けし、ユニット間で連想処理の並列動作を行わせようとする。

本論文の目的は、意味ネットワークマシンＩＸＭ の構成を紹介し、連想メモリを用いた連想処理性能、特に並列演算性能を具体的に示すことにある。さらに言えば、連想メモリ上の演算操作が、超並列処理へのアプローチとして有効であることを示すことにある。

以下では、２章で並列連想記憶のアーキテクチャ、３章で意味ネットワークマシンＩＸＭの構成、４章で連想メモリ上の並列演算性能について述べる。

2．並列連想記憶のアーキテクチャ

図１は、連想メモリの基本的な機能を示している。連想メモリの第１の機能は、並列検索機能、すなわち、キーとなるビットパターンを並列に検索して、一致した語に、フラグを立てる機能である。これに加えて、最近の連想メモリには、並列書き込みの機能が備えられている。これは、フラグが立っている語に、データレジスタの内容を並列的に書き込む機能である。これら二つの機能を複数回組み合わせて用いることによって、種々の連想並列処理が可能となる。例えば、集合演算、並列ハミング距離計算、高速フーリエ変換、リスト処理、コンピュータグラフィックス、画像認識等、いくらでもあげることができる。

一方、連想メモリによる処理には、意味ネットワークにおけるマーカー伝搬や双方向グラフにおける接続性検査などの処理に対して、連想メモリによるＳＩＭＤ処理が必ずしも有効でなく、順次処理にならざるを得ない部

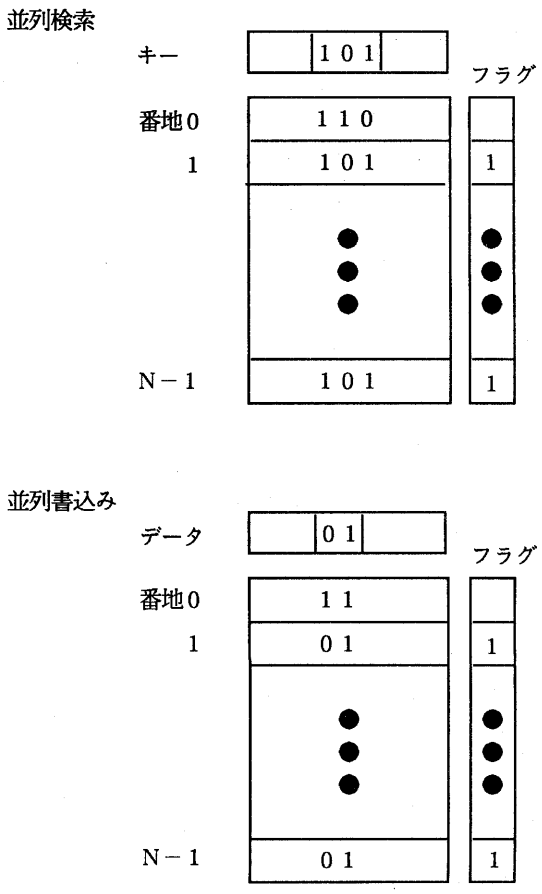


図1　連想メモリの機能

分が存在し、それが連想メモリによる処理のボトルネックになるという問題点も存在している。そのような問題を解決するために、意味ネットワークマシンＩＸＭにおいて、並列連想記憶構成の採用を考えてきている。

並列連想記憶構成では、意味ネットワークにおけるマーカー伝搬に対して等価ノード概念を導入することにより、多重マッチを複数の連想プロセッサユニット（連想メモリを備えたプロセッサ）上に分割割り付けし、ユニット間で連想処理の並列動作を行わせる。

図２は、ＩＸＭにおける並列連想記憶の構成を概念的に示す構成図である。Ｎ語の連想メモリが備えられ、トランスピュータで制御されるユニットがｎ台あり、それらがマーカー等を伝搬するための高機能ネットワークに接続され、お互いに通信できるようになっている。

並列連想記憶アーキテクチャを考える上で、処理すべき問題の特性に依存して解決されなければならない課題がある。ハードウェア量を一定と仮定すると、図３に示すように、ＰＥ数とＰＥ当たりの連想メモリ容量との関係は反比例する。また、ＰＥ並列処理能力を大きくする（＝グレインサイズを小さくして、ユニット数を増やす）ことによる利益と、それに伴って発生するユニット間の通信オーバヘッドによる不利益の間のバランスも考える必要がある。このように、連想並列度を主に考えるかＰＥ並列度を主に考えるかの、問題が持つ並列性の内容に依存する設計方針は、これから明確にしていかなければならない点である。

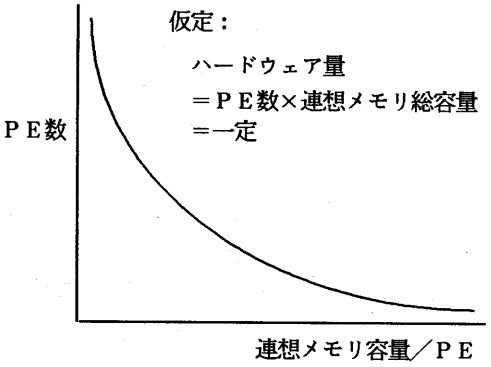


図３　連想並列処理度とＰＥ並列処理度の関係

３．意味ネットワークマシンＩＸＭの構成

意味ネットワークマシンＩＸＭは、並列連想記憶構成の一例であるととらえることができる。ＩＸＭプロトタイプ［7-8］の試作を経て、現在のところ、そこでの問題点を改良し高速化したＩＸＭ２［9］の完成後のハードウェア／ソフトウェアの調整段階にある。

ＩＸＭ２では、完全結合の８個の連想プロセッサ（ＰＥ）と、それらＰＥすべてに接続路を有するネットワークプロセッサ（ＮＰ）を基本グループとして、その基本グループ間が８個のＮＰ同士で完全結合され、それらのＮＰすべてに接続路を有するＮＰを経由してさらに上位のグループと接続される構成を採用している。

図４は、トランスピュータと４Ｋ語（１語４０ビット）の連想メモリを備えるＰＥ６４台（０番から６３番まで）と、トランスピュータと２Ｋ語の連想メモリを備え、ブロードキャスト機能を持つＮＰ９台（６４番から７２番まで）とから成るＩＸＭ２の接続形態を示している。

７２番のＮＰは、図５の右側に示されるホストのＳＵＮ－３／２６０のＶＭＥバスとの間のデュアルポートメモリを備えるインタフェースボードに接続される。このようなローカルな完全結合をベースとした階層的接続方式は、完全結合による通信オーバヘッドの軽減と大規模化の容易さを特長としている。

ＩＸＭ２における連想メモリの総容量は、２５６Ｋ語（連想処理用）＋１

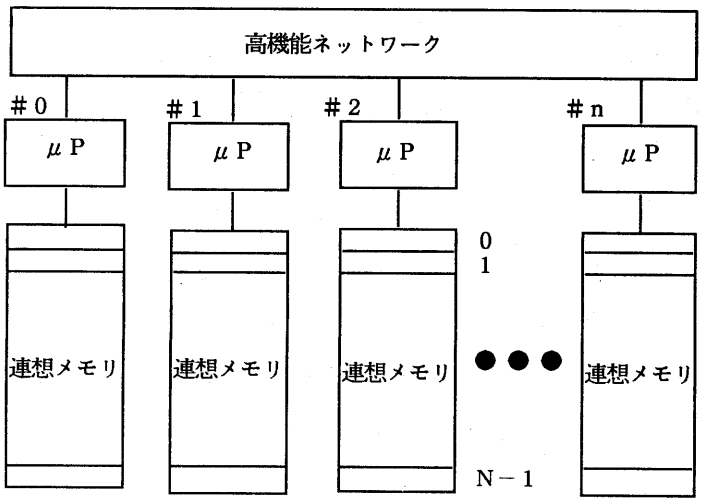


図２　並列連想記憶の概念構成

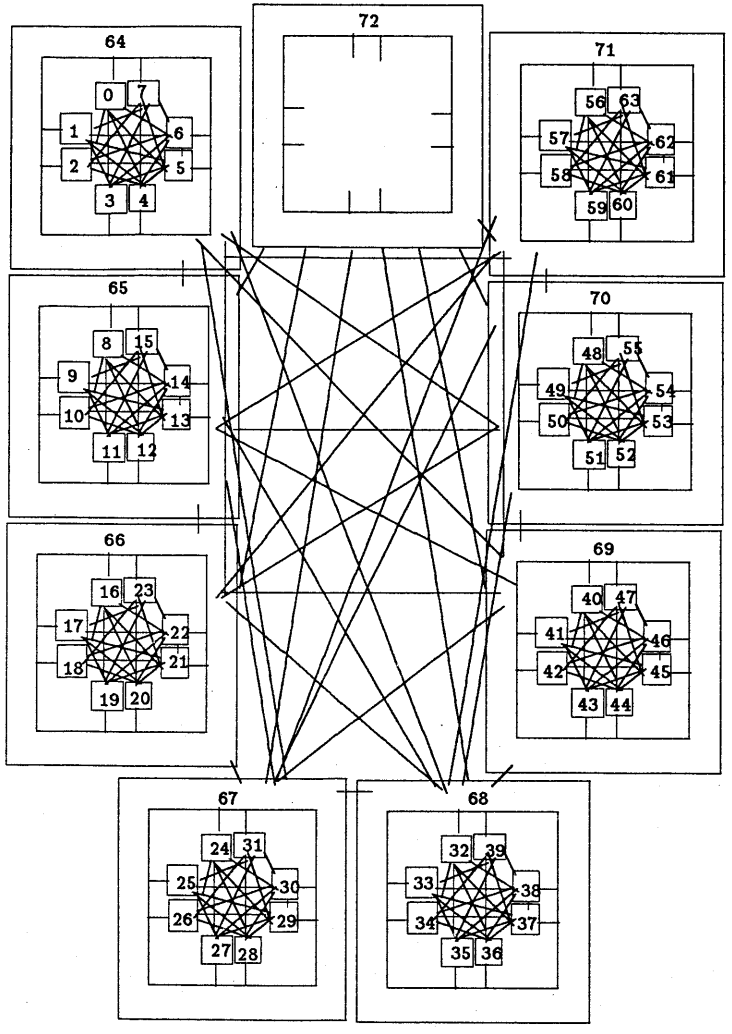


図4　ＩＸＭ２のネットワーク構成

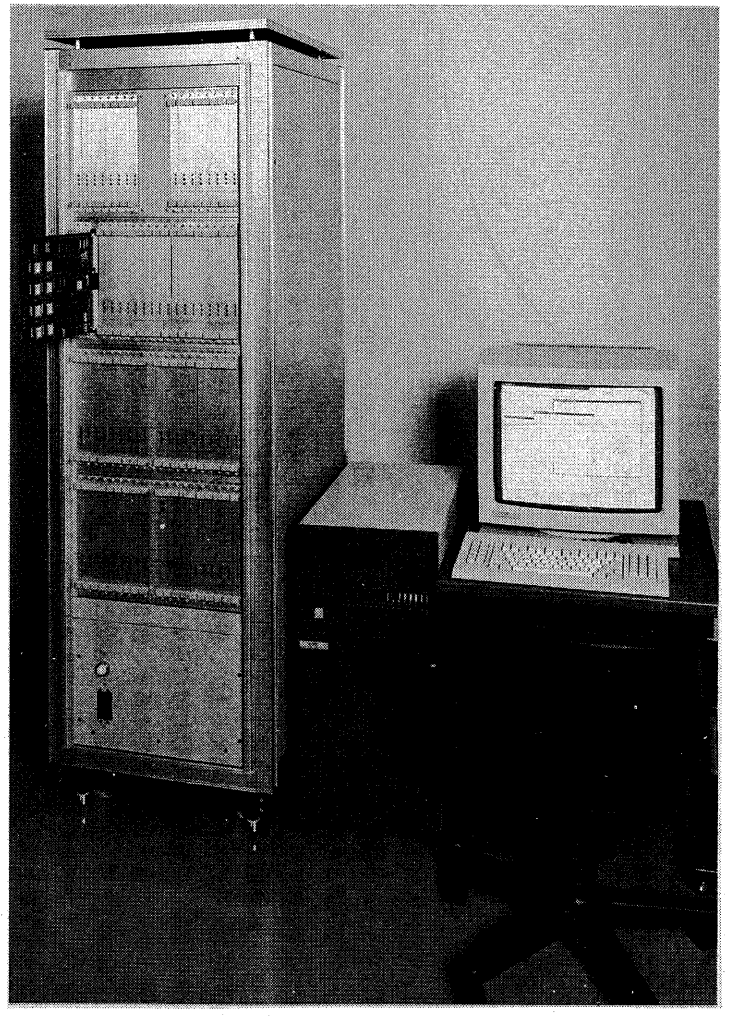
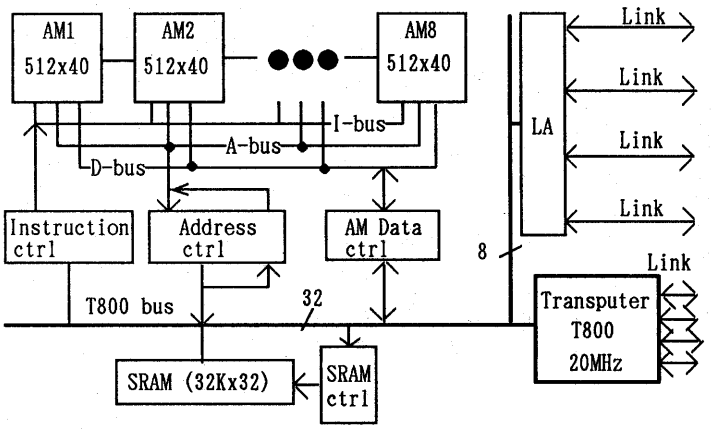


図5　ＩＸＭ２の外観写真



AM： Associative Memory　　D-bus： Data bus
I-bus： Instruction bus　　LA： Link Adaptor
A-bus： Address bus

図6　ＰＥボードのブロック図

８Ｋ語（マーカールーティング用）まで実装可能である。図6のブロック図に示すように、ＰＥボード当たりの連想メモリ容量は、５１２語×４０ビット構成の連想メモリチップ（ＮＴＴ提供）[3] を８個実装するので、４Ｋ語である。

１語の構成は図7に示すように、４バイトと８ビットタグから成る。また各語毎に、検索でフラグが立ったかどうかを示す１ビットのＳレジスタと、その語が検索対象となりうるかを制御するガーベージフラグが付随する。

１語内の４バイトはバイト単位で書き込みが可能である。また８ビットタグは各タグごとに書き込みが可能である。したがって、連想メモリへの書き込みマスクレジスタは１２ビットである。一方、検索は、検索マスクをセット後、一語４０ビットを基本に行う。従って検索マスクレジスタは４０ビット長ある。

ＩＸＭ２で用いるトランスピュータは、３２ビットプロセッサであるので、連想メモリの１語の０、１、２バイト目と８ビットタグが、トランスピュータの１語３２ビットに対応するようにしている。連想メモリの残りの３バイト目は、８ビットの拡張レジスタを設け、これをトランスピュータがアクセスするようにして、連想メモリの全ビットを利用できるようにしている。

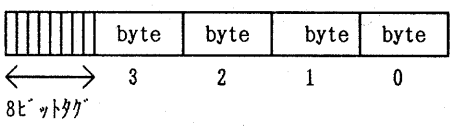


図7　連想メモリチップの一語の構成

連想メモリチップは、２７種の命令モードを備える。それらのうち、２６種の命令は、初期化、読み出し、書き込み、検索、ガーベージコレクションの５つに大別される。これらの機能のうち、ＩＸＭ２の応用プログラム開発において有用なものは、並列書き込みと、複数語にまたがる検索機能である。

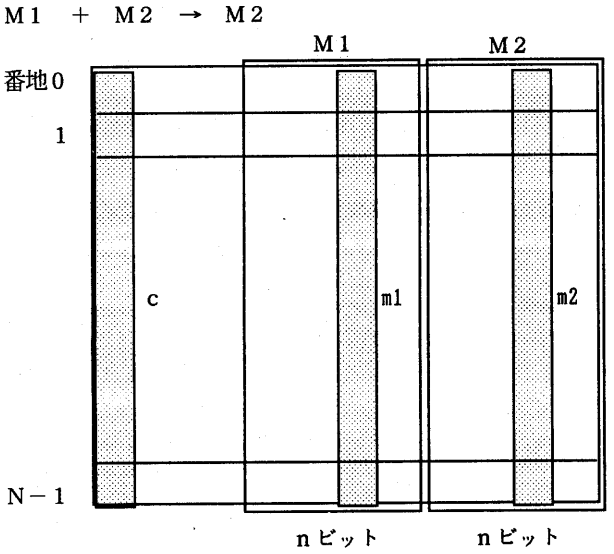
前者の並列書き込みは、ある検索の結果ヒットした複数の語に対して、同時にデータを書き込む機能である。これは、パターンマッチや集合演算において、処理アルゴリズムのオーダを低減する上で有用である。

後者の複数語にまたがる検索機能は、プログラムで用いる基本データが１語４０ビットに収まらず、複数語にまたがる場合にも、連想メモリのＳＩＭＤ的処理を、１語のときと同様に実現することができ有用である。この種の検索機能の実現のために、ある語（アドレスｎ）の上位語（アドレスｎ－１）における検索結果（フラグが立っているかどうか）をその語（アドレスｎ）に対する検索の際に、ＡＮＤかＯＲのいずれかの形で反映させることができる。たとえばＡＮＤならば、上位語の検索フラグが立ち、かつ次の語（ｎ）がヒットした時のみ、その語（ｎ）の検索フラグが立つ。

４．連想メモリ上の並列演算性能

連想メモリ上での並列演算の一例として、並列加算の方法を図８を用いて説明する。加算すべきデータが格納された、Ｎ語の連想メモリを想定する。各語に、Ｍ１フィールド、Ｍ２フィールド、キャリーを格納するための１ビットフィールドを用意する。Ｎ語すべてについて Ｍ１フィールドの内容とＭ２フィールドの内容を、並列的に加算して、Ｍ２フィールドに格納することを考える。具体的には、Ｍ１とＭ２の各フィールドから１ビットずつ切り出してきて、ビットシリアルで計算することを考える。

加算前と加算後のビットパターンの変化を整理すると、図中の表のようになる。加算前は、キャリーまで含めると３ビットのパターンであるから、８通りの組み合わせ



ビットパターン

加算前			加算後		
c	m2	m1	c	m2	
1	1	1	1	1	
1	0	0	0	1	if "100", write "01".
1	0	1	1	0	
1	1	0	1	0	if "110", write "10".
0	1	1	1	0	if "011", write "10".
0	1	0	0	1	
0	0	1	0	1	if "001", write "01".
0	0	0	0	0	

図8　連想メモリ上の並列加算

が存在するが、そのうち４通りは加算の前後で変化しない組み合わせであり、変更操作が不要である。加算の前後で変化する残りのビットパターンでは、ビットパターンの変更操作を行う必要がある。

ビットパターンの変更は、図１に示した連想メモリの機能、すなわち並列検索と並列書き込みを用いて行う。まず、“１００”というビットパターンを並列検索し、一致した語にフラグを立てる。次に、フラグが立てられた語に“０１”というパターンを並列書き込みする。その他の３通りの組み合わせに対しても、同様に操作する。これで、切り出された１ビットの加算が行われたことになる。フィールド幅はｎビットであるから、ビット位置のマスキングをシフトしながら、そのような１ビット加算をｎ回実行すれば、並列加算が行われたことになる。これを以下では、従来アルゴリズムと呼ぶ。

連想メモリの同一語中の二つのフィールドに置かれた値を並列掛算するには、キャリーフィールドと結果を格納する積フィールドを用意して、乗数フィールドの各ビットにもとづいて、被乗数フィールドの内容と積フィールドの内容との間で、シフト加算を繰り返すことによって行う。シフト加算は、シフトして加算する操作であり、上記の並列加算においてたしこむ際に、積フィールドの検索および書き込み対象のビット位置を、ずらして操作することによって実現される。

ＩＸＭにおける連想メモリでは、通常のメモリと同様にアドレス指定で書き込みおよび読み出しができる他に、図９に示す基本操作が可能である。実際のプログラムでは図１０に示すように、検索マスクパターンと書き込みマスクパターンを各々のレジスタにセットした後、並列検索と並列書き込みを行う。並列書き込み後に変更したビットパターンを再び検索しないような順序にし、同一ビットパターンを２回続けて並列書き込みする代わりに、AM. ＯＲ. ＳＥＡＲＣＨ（同一語における新たな検索結果と、検索フラグビットの内容との間でＯＲをとる）操作を用いて、並列書き込みを一回減らすようにする。ステップ数としては、１ビットあたり９ステップの操作が必要になる。これを以下では、改良アルゴリズムと呼ぶ[10]。

基本操作	意味
AM. SEARCH	検索し、一致ワードにフラグを立てる
AM. AND. SEARCH	検索し、フラグとのＡＮＤをとる
AM. OR. SEARCH	検索し、フラグとのＯＲをとる
AM. LAND. SEARCH	検索し、上位フラグとのＡＮＤをとる
AM. LOR. SEARCH	検索し、上位フラグとのＯＲをとる
AM. PAR. WRITE	フラグがオンのワードに並列に書込む
AM. SEARCH. MASK	検索マスクパターンをセットする
AM. WRITE. MASK	書き込みマスクパターンをセットする

図9　連想メモリに対する基本操作

ｍ１とｍ２のビット位置を変えながら、以下の操作をｎ回行う。

```
AM.SEARCH.MASK
AM.WRITE.MASK
AM.SEARCH (c=1,m1=0,m2=0)
AM.PAR.WRITE (c=0,m2=1)
AM.SEARCH (c=1,m1=0,m2=1)
AM.OR.SEARCH (c=0,m1=1,m2=1)
AM.PAR.WRITE (c=1,m2=0)
AM.SEARCH (c=0,m1=1,m2=0)
AM.PAR.WRITE (c=0,m2=1)
```

図１０　フィールド間並列加算の改良アルゴリズム

ここで、ｃはキャリー用の１ビット、ｍ１、ｍ２は各々、Ｍ１、Ｍ２フィールド中の対応する１ビットを示す。

図１１　連想メモリによる並列加算とトランスピュータによる順次加算の比較

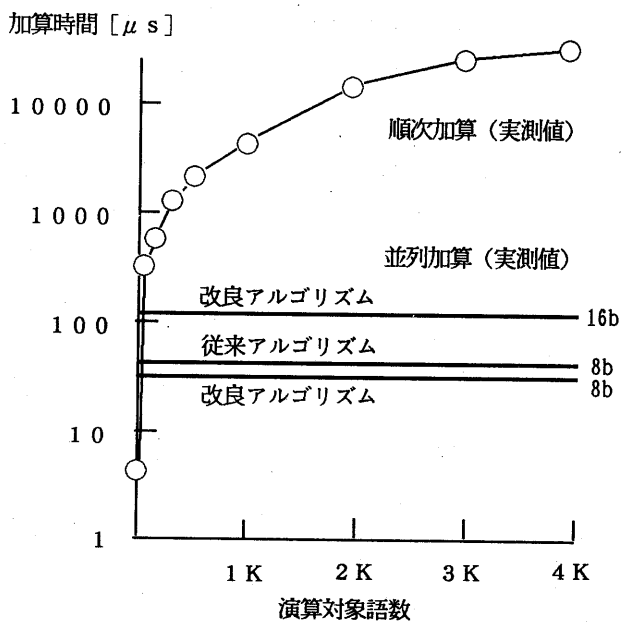


図１１は、各語中の二つのデータを加算する場合に、トランスピュータによる順次加算の演算時間が語数に比例するのに対して、連想メモリによる並列加算では演算時間が一定であることを示している。すなわち、トランスピュータによる順次加算時間は語数に比例するので、縦軸の目盛りに対応して対数曲線になり、連想メモリによる並列加算時間は、語数に依存しないので直線になる。

図１１を描く際に実測した結果の一部を数値で示せば、次のようになる。但し、トランスピュータのクロックは２０ＭＨｚ、ＳＲＡＭのアクセス時間は２００ｎｓ、連想メモリは５ＭＨｚで動作し、そのアクセス時間は平均３２５ｎｓである。プログラムはすべてＯＣＣＡＭ２を用いて記述され、目算しやすいように演算対象語数を１Ｋ語に設定した。演算時間はＯＣＣＡＭ２からアクセスできるタイマを用いて計測された。

トランスピュータによる順次加算時間は、ＳＲＡＭの１Ｋ語の各語中の８ビットデータ同士の加算では、６０６１μｓ（プログラムオンチップ時）または５１１５μｓ（プログラムオフチップ時）である。連想メモリの１Ｋ語の各語中の８ビットデータ同士の加算では、トランスピュータによる順次加算時間が５５３０μｓ（プログラムオンチップ時）または５７３５μｓ（プログラムオフチップ時）であるのに対して、連想メモリによる並列加算時間は、従来アルゴリズムでは５１μｓ、ＡＭ．ＯＲ．ＳＥＡＲＣＨ操作を用いて、並列書き込みを一回減らした改良アルゴリズムで４６μｓである。

したがって、１Ｋ語の各語中の８ビットデータ同士の加算では、トランスピュータによる順次加算（プログラムオンチップ時）と連想メモリによる並列加算（改良アルゴリズム）の比は、約１２０倍にもなる。

１６ビットデータに対しては、連想メモリの１語で処理できるビット数が１２ビットまでという制約から、１６ビットを上位と下位に分けて２語に格納する必要がある。そのようにして１６ビットデータ同士を加算する場合、上位語への桁上げ操作を必要とする。桁上げ操作は、上位の語でキャリーが立っているものを

検索し、次に下位の語でその前の語の検索フラグが立っているものを、AM. LAND. SEARCHを用いて検索する。検索フラグが立った語に、キャリーを書き込む。それに２３μｓを要するので、１１５μｓ（＝４６×２＋２３）である。

５．おわりに

並列連想記憶のアーキテクチャ、意味ネットワークマシンＩＸＭの構成、連想メモリ上の並列演算性能について述べた。連想メモリによるフィールド間並列演算は、対象となる語数が多く、フイールドあたりのビット数が少ないほど有利である。具体的には、順次加算の曲線が並列加算の直線との交点よりも大きい語数を対象とする場合に、連想メモリによる並列加算が有利であり、しかも、語数が大きくなればなるほど、オーダがひらいてくる点は極めて重要である。

ＩＸＭ２では８ビットの並列加算の場合、順次加算と比べて、連想メモリ４Ｋ語実装のＰＥあたり約４８０倍の性能向上が得られることが実証されている。通信オーバヘッドを考慮しなければ、ＩＸＭ２全体で順次加算の約3×10^4（＝約４８０×６４）倍の性能向上が得られることになる。これらのことは、種々の応用において、超並列処理を実現しようとする際に、ＰＥ並列処理方式の検討と共に、連想メモリによる並列処理の検討を行うべきであることを示していると考えられる。

謝辞

連想メモリについて援助して下さったＮＴＴ－ＬＳＩ研究所小倉武氏、本研究の機会を与えられた電子技術総合研究所柏木寛所長、棟上昭男情報アーキテクチャ部長に感謝する。

参考文献

[1] T. Ogura, S. Yamada and T. Nikaido, "A 4-kbit Associative Memory LSI", IEEE Journal of Solid-State Circuits, Vol. SC-20, No. 6, pp.1277-1282, 1985.

[2] H. Kadota, J. Miyake, Y. Nishimichi, H. Kudo and K. Kagawa, "An 8-kbit Content-Addressable and Reentrant Memory", IEEE Journal of Solid-State Circuits, Vol. SC-20, No. 5, pp.951-957, 1985.

[3] 小倉武、山田慎一郎、山田順三、長沼次郎、丹野雅明、"20Kb CAM (Content Addressable Memory)"，電子情報通信学会技術研究報告、Vol.87, No.349, pp.31-37, CPSY 87-33, 1988.

[4] C. C. Foster, "Content Addressable Parallel Processors", Van Nostrand Reinhold Company, New York, 1976.

[5] L. Chisvin and R. J. Duckworth, "Content-Addressable and Associative Memory: Alternatives to the Ubiquitous RAM", Computer Vol.22, No.7, pp. 51-64, July 1989.

[6] I. D. Scherson and S. Ilgen, "A Reconfigurable Fully Parallel Associative Processor", Journal of Parallel and Distributed Computing, Vol.6, No.1, pp.69-89, February 1989.

[7] 樋口哲也、古谷立美、楠本博之、半田剣一、国分明男、"意味記憶システムＩＸ（イックス）のプロトタイプについて"、情報処理学会研究報告、知識工学と人工知能 54-8、1987.

[8] T. Higuchi, T. Furuya, H. Kusumoto, K. Handa and A. Kokubu, "The Prototype of a Semantic Network Machine IXM", Proc. of 1989 International Conference on Parallel Processing, Vol.1, pp.217-224, 1989.

[9] 樋口哲也、古谷立美、楠本博之、半田剣一、国分明男、"意味ネットワークマシンＩＸＭ第２版の概要"、情報処理学会研究報告、計算機アーキテクチャ 78-3、1989.

[10] 国分明男、樋口哲也、古谷立美、“ＩＸＭにおける連想メモリ上の並列演算性能”、情報処理学会第３８回（昭和６４年前期）全国大会、pp.1575-1576、1989.